Processor – Sharing

Can assign a fraction of a machine to a job. If we assign $\beta$ of a machine to a job for $t$ time units, then the job has executed $\beta t$ units of time

Ex: P2 | prmp, prec | Cmax

Note: At time 0, J1, J2, and J3 are at level 5.
At time 1.5, and J3 and J5 are at level 4.
At time 2.5, J4, J5 and J6 are at level 3.
Muntz – Coffman Algorithm

Assign one machine each to the jobs at the highest level. If there is a tie among b jobs (because they are at the same level) for the last a (a < b) machines, then assign a/b of a machine to each of these b jobs whenever either of the two events described below occurs, reassign the machines to the unexecuted portion of the jobs accordingly to the above rule. There are

Event 1: A job is completed

Event 2: We reach a point where, if we were to continue the present assignment, we would be executing some jobs at a lower level at a faster rate than other jobs at a higher level.

Ex: P | prmp, intree | Cmax

( Leave it as exercise )
**List Scheduling** (Non-Delay schedule)

1. The set of jobs is ordered accordingly to some priority scheme and a list of such jobs is formed.
2. Whenever a machine becomes free for assignment, the list is scanned in a fixed direction and the first unexecuted ready job encountered in the scan is assigned to the machine.

**Note:**

1. List scheduling in a subclass of the nonpreemptive scheduling discipline.
2. List scheduling is not as effective as nonpreemptive scheduling.
List scheduling exhibits certain anomalous behavior.

The length of a list schedule depends on processing times of the jobs, the number of machines, the precedence constraints, and the priority list.

Increasing the no. of machines
Decreasing the processing time
Removing some precedence constraints
Changing the priority list

\[
L = (J_1, J_2, J_3, J_4, J_5, J_6, J_7, J_8)
\]

\[
m = 3
\]

1. Increase m by 1

\[
L = (J_1, J_2, J_3, J_4, J_5, J_6, J_7, J_8)
\]

\[
m = 3
\]
2. Decrease each processing time by 1.

\[
\begin{array}{cccc}
2 & 5 & 8 & 13 \\
J_1 & J_5 & J_8 & \\
J_2 & J_4 & J_6 & J_9 \\
J_3 & J_7 & & \\
\end{array}
\]

3. Remove the edges (J_4, J_5) and (J_4, J_6)

\[
\begin{array}{cccc}
3 & 4 & 7 & 8 & 16 \\
J_1 & J_6 & J_9 & \\
J_2 & J_4 & J_7 & \\
J_3 & J_5 & J_8 & \\
6 & 10 & & \\
\end{array}
\]

4. Change the list to \( L^1 = (J_1, J_2, J_4, J_5, J_6, J_3, J_9, J_7, J_8) \)

\[
\begin{array}{cccc}
3 & 5 & 14 \\
J_1 & J_3 & J_9 & \\
J_2 & J_5 & J_7 & \\
J_4 & J_6 & J_8 & \\
\end{array}
\]

**Comparison among List, Nonpreemptive and Preemptive**

**Thm:** Let \( J \) be a set of jobs to be scheduled on machines. Let \( W_L(m) \), \( W_{NP}(m) \) and \( W_P(m) \) denote the optimal makespans accordingly to List, Nonpreemptive and Preemptive scheduling disciplines, respectively. Then we have

\[
\begin{align*}
W_L(m)/W_{NP}(m) & \leq 2 - \frac{1}{m} \\
W_L(m)/W_P(m) & \leq 2 - \frac{1}{m} \\
W_{NP}(m)/W_P(m) & \leq \frac{2m}{m+1}
\end{align*}
\]